POINT CLOUD REFINEMENT WITH A TARGET-FREE INTRINSIC CALIBRATION OF A MOBILE MULTI-BEAM LIDAR SYSTEM

Houssem Nouira, Jean-Emmanuel Deschaud, Francois Goulette

POINT CLOUD REFINEMENT WITH A TARGET-FREE INTRINSIC CALIBRATION OF A MOBILE MULTI-BEAM LIDAR SYSTEM

H. Nouira[a], J. E. Deschaud[a], F. Goulette[a]

[a] MINES ParisTech, PSL - Research University, CAOR - Center for Robotics, 60 Bd St-Michel 75006 Paris, France - (houssem.nouira, jean-emmanuel.deschaud, françois.goulette)@mines-paristech.fr

Commission III, WG III/2

KEY WORDS: LIDAR, Intrinsic, Calibration, Mobile Mapping, 3D, Automatic, Computer Processing, Velodyne

ABSTRACT:

LIDAR sensors are widely used in mobile mapping systems. The mobile mapping platforms allow to have fast acquisition in cities for example, which would take much longer with static mapping systems. The LIDAR sensors provide reliable and precise 3D information, which can be used in various applications: mapping of the environment; localization of objects; detection of changes. Also, with the recent developments, multi-beam LIDAR sensors have appeared, and are able to provide a high amount of data with a high level of detail.
A mono-beam LIDAR sensor mounted on a mobile platform will have an extrinsic calibration to be done, so the data acquired and registered in the sensor reference frame can be represented in the body reference frame, modeling the mobile system. For a multi-beam LIDAR sensor, we can separate its calibration into two distinct parts: on one hand, we have an extrinsic calibration, in common with mono-beam LIDAR sensors, which gives the transformation between the sensor cartesian reference frame and the body reference frame. On the other hand, there is an intrinsic calibration, which gives the relations between the beams of the multi-beam sensor. This calibration depends on a model given by the constructor, but the model can be non optimal, which would bring errors and noise into the acquired point clouds. In the litterature, some optimizations of the calibration parameters are proposed, but need a specific routine or environment, which can be constraining and time-consuming.
In this article, we present an automatic method for improving the intrinsic calibration of a multi-beam LIDAR sensor, the Velodyne HDL-32E. The proposed approach does not need any calibration target, and only uses information from the acquired point clouds, which makes it simple and fast to use. Also, a corrected model for the Velodyne sensor is proposed.
An energy function which penalizes points far from local planar surfaces is used to optimize the different proposed parameters for the corrected model, and we are able to give a confidence value for the calibration parameters found. Optimization results on both synthetic and real data are presented.

1. INTRODUCTION

Light Detection and Ranging (LIDAR) sensors are useful for many tasks: mapping (Nuchter et al., 2004), localization (Narayana K. S et al., 2009) and autonomous driving (Grand Darpa Challenge, 2007) are some of the tasks where LIDAR sensors are useful. Multi-beam LIDAR sensors give data with a high density of points and are more precise than mono-beam sensors: they are also evolving fast, and become cheaper with time. To give accurate data, multi-beam sensors have an intrinsic calibration which needs to be done: generally, this calibration depends on the geometric disposition of the beams in the sensor. The calibration follows a model, which is given by the constructor: the model can be corrected in order to give more precise data.
The different representations of each acquired point are illustrated with figure 1, with the different reference frames. The intrinsic calibration describe the transformation of the acquired from spherical coordinates to cartesian coordinates, referenced in the same reference frame. The optimization we are speaking of consists in finding some additionnal parameters for each beam of the LIDAR sensor. A beam is set as a reference, and we optimize the intrinsic calibration parameters of the other beams regarding this reference. The procedure is described with more details in section 3..
In this article, we will call calibration of the sensor the intrinsic calibration of the multi-beam LIDAR sensor: the intrinsic calibration of the sensor allows to have data correctly referenced in the cartesian sensor reference frame. The solution we propose is,

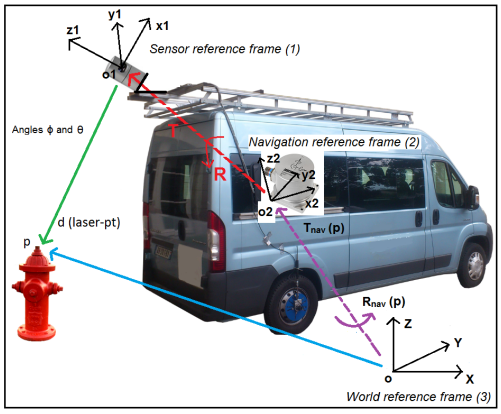

Figure 1. Geo-referencement of the data

after the acquisition, to estimate the parameters of the calibration that give the ”best” - depending on some criteria - point cloud. We present an unsupervised calibration method for multi-beam LIDAR sensors, which does not need any calibration target.

This paper is organized as follow: in section 2., we present the state of the art concerning the algorithms for the intrinsic calibration of multi-beam LIDAR systems. Section 3. presents our optimization methods for the intrinsic calibration parameters. Section 4. shows some experimentation results obtained with our algorithm. Finally, section 5. finally gives a conclusion to this paper.

2. RELATED WORK

Figure 1 shows our mobile mapping system, with a LIDAR sensor mounted on the roof which is the Velodyne HDL-32E: we give its specificities in section 3.. The figure also gives the different representations of an acquired point by the mapping system.

- Raw data are acquired by the multi-beam sensor mounted on the Vehicle, which are the distance of the point acquired to the sensor and two angles.
- The raw data can be expressed in the Cartesian reference frame of the sensor: this is done using the intrinsic calibration parameters of the sensor.
- The extrinsic calibration gives the geometric transformation between the sensor and the IMU - which are mounted on the mobile platform - and is needed to have coordinates registered in the navigation reference frame. There are six parameters to retrieve, three rotations and three translations.
- The data can also be geo-referenced by applying the transformation between the navigation reference frame and the world reference frame to these data. This transformation is given by the fusion of data from many sensors embedded on the vehicle, such as an IMU and a GPS.

The calibration of a LIDAR sensor is an important task, whether it has many beams or not. It allows the sensor to give correctly referenced data during the process of acquisition, which is necessary for many tasks, such as point clouds segmentation (Serna and Marcotegui, 2014) for example. In this section, we will talk about some of the intrinsic calibration techniques for multi-beam LIDAR acquisition systems.

Multi-beam LIDAR sensors can be separated into two categories:

- Sensors made of several mono-beam LIDAR sensors, for which the data are fusionned and which provide 3D information with a specific calibration routine, such as the riegl sensor (Riegl LIDAR sensor datasheet, 2015).
- Multi-beam LIDAR sensors such as the Velodyne (Velodyne site web, 2015) or the quanergy (Quanergy product page, 2015).

Because mono-beam LIDAR sensors may be cheaper than multi-beam, some 3D mapping system are constructed around several mono-beam sensors. These sensors also need to be calibrated, and some automatic algorithm exist. This is for example the case in (Sheehan et al., 2012), where the authors propose an automatic method for the self-calibration of a 3D-laser. The 3D laser is made of 3 mono-beam LIDARs SICK LMS-151 placed on a rotating plate, and for the self-calibration of the sensor, he measures the quality of the acquired point clouds and corrects the calibration parameters in consequence.
In (Lin et al., 2013), another automatic optimization for the calibration of a self-made multi-layer LIDAR sensor is proposed. He mounted a single-layer HOKUYO UTM-30LX LIDAR sensor on a pan-tilt unit, and estimated the new parameters induced by the pan-tilt unit by correcting the structure of planar surfaces which were not correctly planar with bad calibration parameters.

In this section, we will talk about the existing work on the intrinsic calibration of Velodyne sensors, first because these sensors are widely popular since 2007, but also because this is the kind of sensor we used for our experimentations. The Velodyne sensors appeared recently - around the year 2007 -, but we already can find some calibration techniques which are specific to this kind of sensor. Indeed,(Glennie and Lichti, 2010) and (Muhammad and Lacroix, 2010) propose an optimization of the intrinsic parameters for the 64-beam version, and (Chan and Lichti, 2013) proposes an intrinsic calibration for the 32-beam model. In (Glennie and Lichti, 2010) and (Muhammad and Lacroix, 2010), the authors use a particular calibration environment to optimize the intrinsic parameters, which contains many planar walls: these walls are extracted from the acquired point cloud, and their structure is corrected in order to optimize the calibration parameters. In (Chan and Lichti, 2013), the optimization of the intrinsic parameters is done statically, by using environment information such as planar wall and vertical cylinders. In (Chan and Lichti, 2015), the authors propose an extension of the method presented in (Chan and Lichti, 2013): they also correct the intrinsic calibration parameters in a kinematic mode, by correcting planar walls and culinders extracted from the point clouds.
In (Huang et al., 2013), the authors propose a full extrinsic calibration of a system made of a Velodyne 64 beams LIDAR sensor and an infra-red camera: they also optimize some intrinsic calibration parameters of the LIDAR sensor. They use a calibration target, and with the infra-red images, they have the impacts of the LIDAR sensor on the target. In (Atanacio-jiménez et al., 2011), the authors present an automatic algorithm to optimize the intrinsic and extrinsic parameters of a Velodyne HDL-64E sensor. A corrected model for the intrinsic parameters is proposed and the parameters are optimized to fit the model. All the optimization are done by using a calibration target.
Finally, there is also another optimization for the intrinsic calibration parameters which is proposed in (Levinson and Thrun, 2010). For the optimization, the authors defined an energy function which penalizes points that are far away from planar surfaces extracted from the acquired data. For the intrinsic optimization, the authors start from an initial estimate, and iteratively compute values of their energy function by modifying the concerned intrinsic parameters in the neighborhood of the initialization. They use a grid search to optimize the parameters and reduce the size of the neighborhood at each iteration. The main problem is that the minimization can be long if a high precision is required. Also, because the neighborhood is a discrete space, it is possible not to reach the optimal solution.

To optimize the calibration parameters of the multi-beam sensor, we use an energy function which only needs information extracted from the acquired point clouds. No calibration target is used, and the process is unsupervised. The defined energy function is also minimized iteratively, as it is explained in section 3.. However, the differences with respect to existing methods are manyfold:

- First, the energy is defined as the sum of the squared distance of each points to the closest plane it should belong to, and its expected optimal (minimum) value is related to the global covariance of the point cloud noise.
- We also introduce in the energy weights which exploit the local planarity of data.
- Our method leads to a more accurate calibration for the point cloud, and does not need a precise initialization.
- The numerical resolution is faster than existing methods, and is done in acceptable times.
- We give an analysis of the precision obtained for the calibration parameters with the resolution.

3. PROPOSED OPTIMIZATION METHOD

To do our acquisitions, we have a mobile mapping system, presented in figure 1. It is equipped with many sensors to precisely

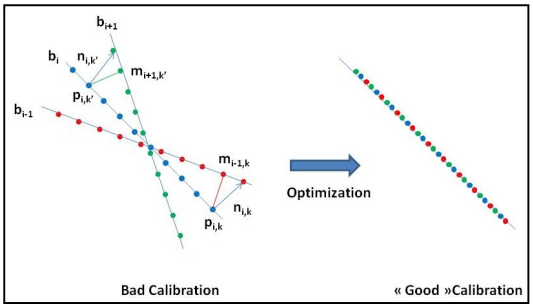

Figure 2. Side-view of a planar surface

get its absolute position during the acquisitions: a BEI DHO5S odometer and an iXBlue LANDINS IMU are used to precisely follow the motion of the vehicle; a Novatel FlexPak 6 GPS is used to retrieve the global position of the vehicle when possible. There is also the multi-beam LIDAR sensor, the 32-beam Velodyne, which is mounted on top of the vehicle, as shown on figure 1. The Velodyne sensor provides up to 700 000 points/s, and covers a vertical field of view of 40° - from -8° to 32° - and an horizontal field of view of 360°. Also, we know the vehicle global pose at each control point, given by the frequence of the fusion IMU+GPS, which is 100 Hz. For our need, we only use the Multi-beam sensor to acquire data, and for geo-referencing the data, we use information given by the proprioceptive sensors: the global position of the vehicle is retrieved by fusioning data from the IMU and the GPS. For the optimization of the intrinsic calibration parameters, we assume that the localization of the vehicle is properly provided by the navigation sensors presented before.

The points in a point cloud come from the combination of the acquisitions of each beam of the multi-beam LIDAR sensor: during the motion of the vehicle, adjacent beams on the sensor will acquire at different times points that belong to the same surface. Figure 2 shows the expected result: with a wrong calibration, points acquired by neighbor beams will not be co-linear, where with a good calibration, lines of points acquired by close beams will overlap. For the optimization of the intrinsic parameters, we suppose that the extrinsic calibration parameters are already optimized: the optimization is done with the algorithm presented in (Nouira et al., 2015), where an optimization method for the extrinsic parameters is detailed.

3.1 Definition of the energy function

To optimize the calibration parameters, we want to consider points which belong to planar surfaces and exploit the previous observation, which is that these surfaces are not exactly planar with a wrong calibration. We start with an initial calibration, and only use information extracted from the point clouds. We do not use any information on the point cloud beforehand, and relie on information obtained during the optimization: no particular data is needed, we suppose that with the density of the LIDAR sensor, points belong to locally planar surfaces. Eq. (1) gives the energy function we defined to optimize the calibration parameters:

$$J(R,T) = \frac{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_{k} w_{i,j,k} * d_{i,j,k}^2(R,T)}{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_{k} w_{i,j,k}} \quad (1)$$

where:

$$\begin{cases} d_{i,j,k}(R,T) = n_{i,k} \cdot (p_{i,k}(R,T) - m_{j,k}(R,T)) \\ p_{i,k}(R,T) = R_{nav}(p'_{i,k}) * (R * p'_{i,k} + T) + T_{nav}(p'_{i,k}) \\ m_{j,k}(R,T) = R_{nav}(m'_{j,k}) * (R * m'_{j,k} + T) + T_{nav}(m'_{j,k}) \end{cases}$$

In equation 1, the other terms are:

- B is a sample of the Velodyne sensor beams, with $B \subset [\![0;31]\!]$
- N is half the number of neighbor beams to beam i taken into account
- k iterates on a subset of the points of beam i
- $w_{i,j,k}$ is a weight, which value is 1 depending on a threshold on the distance between points $p_{i,k}$ and $m_{j,k}$.
- $n_{i,k}$ is the normal at point $p_{i,k}$ to the tangent plane to point $p_{i,k}$.
- $p_{i,k}$ and $m_{j,k}$ are respectively the k^{th} point of beam i, projected in the global reference frame and its nearest neighbor on beam j, also projected in the same reference frame.
- $p'_{i,k}$ and $m'_{j,k}$ are respectively the k^{th} point of beam i, projected in the sensor coordinate frame and its nearest neighbor on beam j, also projected in the same coordinate system.
- R_{nav} and T_{nav} are respectively the rotation matrix and translation vector from the navigation reference frame to the global reference frame. These matrix and vector depend on the time of the acquisition, thus they change from a point to another.
- The energy we defined has a relation to physics: indeed, the energy unit is a square distance (m^2), since it is a sum of squared distances between two points. We suppose that the point cloud contains some noise coming from various sources - motion of the vehicle; errors from the navigation system; errors from the LIDAR sensor -, and that for each point taken into account in the calculation of the energy J, this noise is independent, centered, reduced and follows a normal distribution: with these hypothesis, the energy J follows a chi-squared distribution. Energy J gives an estimate of the variance σ^2 of the point cloud noise when N_t is big enough.

In this section, we will present the optimization of the intrinsic calibration parameters. The optimization of the energy will be detailed.

3.2 Optimization of the calibration parameters

For the intrinsic parameters optimization, we use the energy J defined in (1). Figure 3 gives an illustration of the acquisition of data. Indeed, the Velodyne HDL-32E is composed of 32 beams, which are placed on the same vertical plane. On figure 3 a), there is an example with 2 fibers. The fiber 15 is called "reference": we choose it as a reference because its vertical angle is equal to zero. For each acquisition, the sensor gives the following informations:

- The vertical angle ϕ_i of each beam, regarding the reference fiber.
- The distance $\rho_{i,k}$ between the origin of fiber i and the acquired point k.
- The horizontal angle θ_i, which is introduced by the motion of the sensor.

The intrinsic calibration of the Velodyne 32-beams can be represented with three equations, which transforms the spherical coordinates of each acquired point into cartesian coordinates. The three equations for a point p' acquired by a fiber i at time t are given in equation (2):

$$p'_i(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t)) * cos(\phi_i) \\ -\rho_i(k) * sin(\theta_i(t)) * cos(\phi_i) \\ \rho_i(k) * sin(\phi_i) \end{pmatrix} \quad (2)$$

We want to correct the model for the intrinsic calibration presented in equation (2). Indeed, the model used by the constructor supposes that the sensor is perfect. We choose the following corrected model for each beam, which was presented in (Chan and Lichti, 2013):

- Between each beam, it is supposed that there is the same vertical angle separation. We add an offset $\delta\phi_i$ on each vertical angle ϕ_i to correct little errors which could exist.
- All the beams are supposedly placed on the same vertical plane. An error of alignment can exist, and we add an offset $\delta\theta_i$ on the horizontal angle θ_i.
- We add an offset $\delta\rho_i$ on the distance $\rho_i(k)$ between the origin of beam i and the acquired point k.
- Finally, all the beams are supposed to have the same origin, which is not obvious. We add a little vertical offset for each beam, which takes into account small errors due to differents origins.

All of the offsets we added to equation 2 give a new intrinsic transformation:

$$p'_i(t) = \begin{pmatrix} (\rho_i(k)+\delta\rho_i) * cos(\theta_i(t)+\delta\theta_i) * cos(\phi_i+\delta\phi_i) \\ -(\rho_i(k)+\delta\rho_i) * sin(\theta_i(t)+\delta\theta_i) * cos(\phi_i+\delta\phi_i) \\ (\rho_i(k)+\delta\rho_i) * sin(\phi_i+\delta\phi_i) + H_{z,i} \end{pmatrix}$$

A linearization at the first order gives the following equations for point p'_i:

$$\begin{aligned} p'_{i,k}(t) = {} & p'_{1,i,k}(t) + p'_{2,i,k}(t) * \delta\rho_i + p'_{3,i,k}(t) * \delta\theta_i \\ & + p'_{4,i,k}(t) * \delta\phi_i + p'_{5,i} \end{aligned} \tag{3}$$

where:

$$\begin{cases} p'_{1,i,k}(t) = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t)) * cos(\phi_i) \\ -\rho_i(k) * sin(\theta_i(t)) * cos(\phi_i) \\ \rho_i(k) * sin(\phi_i) \end{pmatrix} \\ p'_{2,i,k}(t) = \begin{pmatrix} cos(\theta_i(t)) * cos(\phi_i) \\ -sin(\theta_i(t)) * cos(\phi_i) \\ sin(\phi_i) \end{pmatrix} \\ p'_{3,i,k}(t) = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t)+90) * cos(\phi_i) \\ -\rho_i(k) * sin(\theta_i(t)+90) * cos(\phi_i) \\ \rho_i(k) * sin(\phi_i) \end{pmatrix} \\ p'_{4,i,k}(t) = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t)) * cos(\phi_i+90) \\ -\rho_i(k) * sin(\theta_i(t)) * cos(\phi_i+90) \\ \rho_i(k) * sin(\phi_i+90) \end{pmatrix} \\ p'_{5,i} = \begin{pmatrix} 0 \\ 0 \\ H_{z,i} \end{pmatrix} \end{cases}$$

Then, for the calculation of these offsets (which are unknown), we first choose a fiber as a reference: this way, we reduce the number of degrees of freedom of the system, which allows us to find a unique solution for these offsets.The fiber chosen as a reference is the fiber 15 of the velodyne, which has a vertical angle of $0°$. We then use the linearization of equation (1) to optimize our intrinsic parameters: in total, there are $4 * 31 = 124$ parameters to optimize. In the equation, these parameters appear in the terms $p'_{i,k}$ and $m'_{j,k}$. We then have a linear least squares problem to solve, with the objective function:

$$S(\delta X_{int}) = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_{k} w_{i,j,k} * (B_{i,j,k} + A_{i,j,k} \cdot \delta X_{int})^2 \tag{4}$$

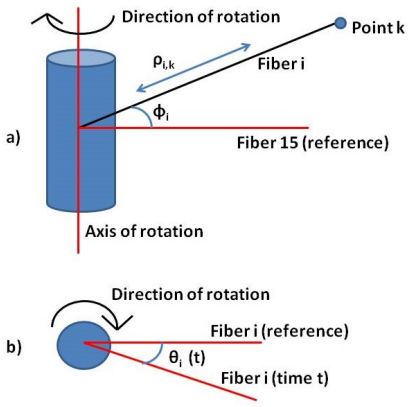

Figure 3. Description of the intrinsic parameters for the Velodyne sensor

where:

$$\begin{cases} \delta X_{int} = ([\delta\rho_i \quad \delta\theta_i \quad \delta\phi_i \quad H_{z,i}]_{i\in[\![0,31]\!]\setminus 15})^T \\ B_{i,j,k=\, n_{i,k}} \cdot \begin{pmatrix} T_{nav}(p'_{i,k}) - T_{nav}(m'_{j,k}) \\ + \left(R_{nav}(p'_{i,k}) * (R(\alpha,\beta,\gamma) * p'_{1,i,k} + T(t_x,t_y,t_z))\right) \\ - \left(R_{nav}(m'_{j,k}) * (R(\alpha,\beta,\gamma) * m'_{1,j,k} + T(t_x,t_y,t_z))\right) \end{pmatrix} \\ A_{i,j,k} = n_{i,k} \cdot \begin{pmatrix} 0_{[\![1,4*i]\!],1} \\ R_{nav}(p'_{i,k}) * R(\alpha,\beta,\gamma) * p'_{2,i,k} \\ R_{nav}(p'_{i,k}) * R(\alpha,\beta,\gamma) * p'_{3,i,k} \\ R_{nav}(p'_{i,k}) * R(\alpha,\beta,\gamma) * p'_{4,i,k} \\ R_{nav}(p'_{i,k}) * R(\alpha,\beta,\gamma) * \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \\ 0_{[\![1+4*(i+1),4*j]\!],1} \\ R_{nav}(m'_{j,k}) * R(\alpha,\beta,\gamma) * m'_{2,j,k} \\ R_{nav}(m'_{j,k}) * R(\alpha,\beta,\gamma) * m'_{3,j,k} \\ R_{nav}(m'_{j,k}) * R(\alpha,\beta,\gamma) * m'_{4,j,k} \\ R_{nav}(m'_{j,k}) * R(\alpha,\beta,\gamma) * \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \\ 0_{[\![1+4*(j+1),124]\!],1} \end{pmatrix} \\ i \; and \; j \neq 15 \end{cases}$$

In eq. (4), we have $i < j$, otherwise the p'_i and m'_j terms are inverted, and $i > 0$ and $j < 31$; if not, the null vectors are not needed.
The solution which minimizes the objective function (4) is the solution of the following linear system:

$$C_{int} \times \delta X_{int} = -V_{int} \tag{5}$$

with:

$$\begin{cases} C_{int} = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_{k} w_{i,j,k} * A_{i,j,k} * A_{i,j,k}^T \\ V_{int} = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_{k} w_{i,j,k} * B_{i,j,k} * A_{i,j,k} \end{cases}$$

3.3 Validity of the calibrations

We defined in the previous sections an energy function that, with an optimization process, should give better calibration parameters for our points clouds. We discuss in this section about the condition that validate a calibration obtained with our optimization process. The value of the energy J should be small enough, under a threshold: as said in section 3.1, our energy follows a chi-squared distribution. A validation threshold at 97% is $3\sigma^2$, with σ^2 the variance of the point cloud noise. For example, for

real data, the noise comes from different sources; with our mobile mapping system, we have a good precision, with a standard deviation for the noise around 5cm. It gives us a threshold of around 75 cm^2 for the value of energy J, in order to validate the calibration process.

We also define an error value for each category of intrinsic parameter, to characterize the difference between each offset of an intrinsic parameter and the associated ground truth when known. The error is the sum of the squares of the final offsets for the intrinsic parameters, which gives:

$$\begin{cases} \Delta\rho = \sum_{i=0/i\neq 15}^{31} (\delta\rho_i)^2 \\ \Delta\theta = \sum_{i=0/i\neq 15}^{31} (\delta\theta_i)^2 \\ \Delta\phi = \sum_{i=0/i\neq 15}^{31} (\delta\phi_i)^2 \\ \Delta H_z = \sum_{i=0/i\neq 15}^{31} (\delta H_{z,i})^2 \end{cases} \quad (6)$$

In the ideal case, these errors should be close to 0 for each parameter. For a real point cloud, these errors should be small.

4. EXPERIMENTAL RESULTS

In this section, we will only present different calibration results on 3 point clouds: 1 is simulated, and 2 come from an acquisition in a real urban area. We tested the optimization on others simulated and real data, and obtained the same results in general.

4.1 Data used for the optimizations

The simulated data used is point clouds which represent an acquisition in a urban area. The environment is made of vertical - to represent walls and façades - and horizontal - the ground, representing the road - planes. This data are used to validate our algorithms: indeed, for this kind of point cloud, we know the ground truth, which are the optimal intrinsic calibration parameters. To validate our optimization, some error is added to each calibration parameter that we want to retrieve, and the expected result is to have the δX as close to zero as possible. The simulated data which is made of 5 million points has the following features: Point cloud #1, which is presented in figure 4, is made of a ground and two vertical planes. The vehicle is doing a turn, and there is no variation of altitude in this point cloud.

The real data come from two different acquisitions, one in the city of Montbeliard, and the other in the city of Dijon, both in France. They are used to show some optimization results on data acquired in different environments:

- Point cloud #2, presented in figure 5 is a point cloud, part of an acquisition in the city of Montbeliard, in France. The point cloud presented contains some turns, several façades and a small variation of altitude. The point cloud is made of 10 million points.
- Point cloud #3, presented in figure 6 is a point cloud, part of an acquisition in the city of Dijon, in France. This time, there is no turn during the acquisition, several façades and a little variation of altitude. The point cloud is made of 5 million points.

4.2 Datasets

In our experiments, we use the same information for both data, simulated and real. We have raw information from the sensor, which is composed of:

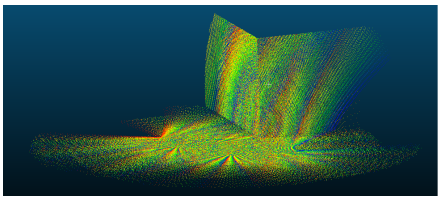

Figure 4. Point cloud #1

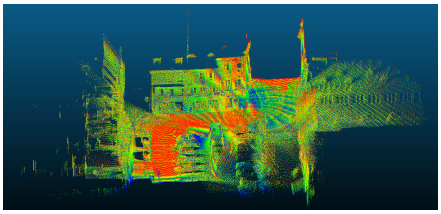

Figure 5. Point cloud #2

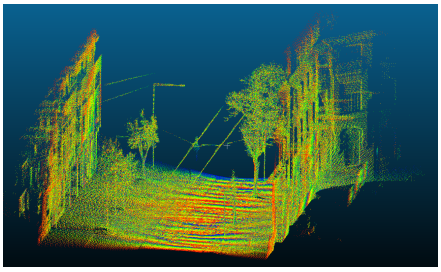

Figure 6. Point cloud #3

- the position and orientation of the vehicle at a frequence of 100 Hz. This is the position of the IMU in the world reference frame, fused with other information from proprioceptive sensors, such as the GPS and the odometer.
- the coordinates of each acquired point in the spherical coordinate system of the sensor reference frame. Since we are optimizing the intrinsic calibration parameters, these data are necessary.
- the ”beam” which acquired each point, since we work with a multi-beam sensor.

These information give us the position of the vehicle and its trajectory with a good observability: indeed, we only work on the calibration parameters of the acquisition system, and to have a well reconstructed point cloud at the end of the optimization, we need to precisely know the trajectory of the vehicle.

4.3 Implementation and algorithm parameters

The algorithm we presented was implemented in C++. The EIGEN library (Eigen library, 2015) was used for all the operations on matrices or vectors, and the FLANN library (FLANN library, 2015) (Fast Library for Approximated Nearest Neighbor) was used for the nearest neighbors search. The different algorithms run on a computer with a Windows 7 - 64 bits OS, 32 GB of RAM and an intel core-i7 processor, with a clock up to 2.80 GHz.
Our algorithm was tested with synthetic and real urban data: for the synthetic data, the parameters were known precisely. For the real data, we have the simplified intrinsic calibration model, and we want to find little biases which correct the model. For both data, we started with initial intrinsic biases, arbitrarily chosen. In our algorithm, we have some parameters to set. We start with subsampling the data about 1 point out of 3, because the point clouds have a high resolution and it reduces the computation times and

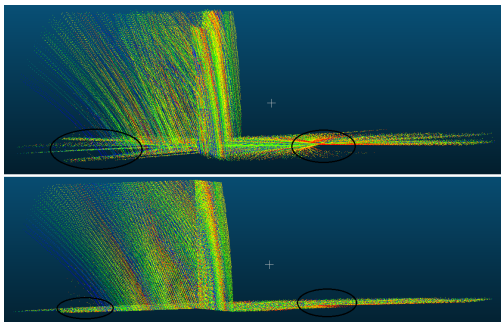

Figure 7. Simulated point cloud #1: top, before optimization of the parameters; bottom, after optimization. The two images have the same point of view.

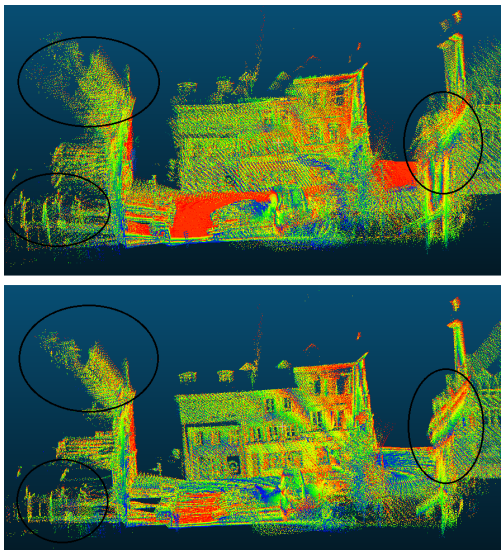

Figure 8. Real point cloud #2: top, before optimization of the parameters; bottom, after optimization. The two images have the same point of view.

the use of memory, without changing the results. The number of neighbor beams for a beam b_i was fixed to 4 (N=2). Concerning the weights $w_{i,j,k}$, a threshold of 20 cm was chosen for the maximal distance d_{max} between a point $p_{i,k}$ and its nearest neighbor $m_{j,k}$ on the neighbor beam.
The parameters were fixed for all the tests which were done: different values were tested, but the ones presented give both good optimization results and computation times.

4.4 Optimization of the intrinsic calibration parameters

In this section, we will present the results of the optimization of the intrinsic calibration parameters as presented in section 3.. We will show the robustness of our optimization method and the improvements on the point cloud.

4.4.1 Results on simulated data. For the simulated data, and because the intrinsic calibration parameters are optimal - this is the way the simulated data are constructed -, we added some biases to the calibration parameters, and compared the results between the optimization of the extrinsic calibration parameters only, and the optimization of all the calibration parameters. The errors added to the intrinsic parameters were between -3 °and 3 °for the angles parameters (ϕ and θ), and between -10 cm and 10 cm for the distance (ρ and H_z). We show the optimization results for point cloud #1; we expect final biases as close to zero as possible.

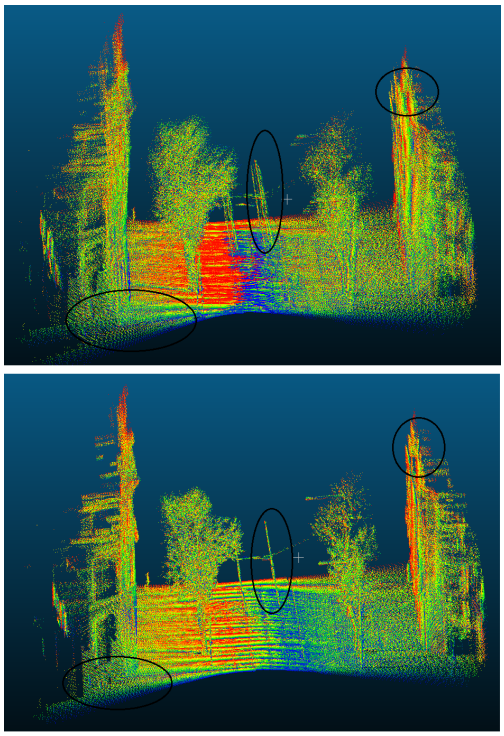

Figure 9. Real point cloud #3: top, before optimization of the parameters; bottom, after optimization. The two images have the same point of view.

Figure 4 presents the same point cloud, on top with bad intrinsic calibration parameters and at the bottom with corrected parameters after our optimization. We see that with the optimization of the intrinsic calibration parameters, we have improved the quality of the point cloud: the plans are correctly planar after the optimization. Figure 10 gives the evolution of the energy through the iterations, and we can see that the energy decreases from a value of 404.58 cm^2 to a value of 0.28 cm^2. Also, we show the evolution of the total weight used to normalize the energy, which has the same evolution as the energy: when the energy decreases, the total weight increases, until the energy converges. It shows that the optimization improves the structure of the point cloud and correctly register the data acquired by the different beams, as explained in section 3.1. Finally, table 11 gives the errors we defined in section 3.3 before optimization, and after optimization of the intrinsic parameters: we can see that the errors are smaller after the optimization, and close to 0 as expected. The final energy is small, and the final intrinsic biases are close to 0, which validate the results of our optimization.
For the computation time of the optimization of the intrinsic calibration parameters, we have for point cloud #1 a computation time of 5 minutes, which is acceptable regarding the high number of parameters optimized.

4.4.2 Results on real urban data. In this section, we will present some results on real urban data. For the real data, we do not know the ground truth: we suppose that the simplified model can be corrected, and expect small biases to add to the model, as presented in section 3.. With our tests, we have seen that there is no visual improvements when there is no intrinsic parameters errors added at the start of the optimization; still, the energy is reduced a little and the final biases for the intrinsic parameters are small, under 10 cm for the translation ones and under 1 degree for the rotation ones. For the presented optimizations results, we added some important errors to the intrinsic parameters - to visu-

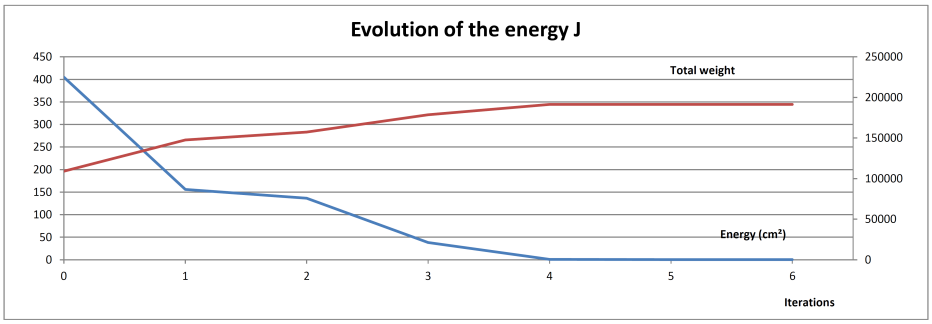

Figure 10. Evolution of the energy of synthetic point cloud #1. In blue, we show the evolution of our energy during the optimization; in red, this is the number of paired points at each iteration for our optimization method

	$\Delta\rho(cm^2)$	$\Delta\theta(^{\circ 2})$	$\Delta\phi(^{\circ 2})$	$\Delta H_z(cm^2)$
Initial errors	3100	193.75	279	3100
Final errors after our optimization	$1.02 * 10^{-2}$	$1.16 * 10^{-5}$	$2.73 * 10^{-4}$	$5.47 * 10^{-1}$

Table 11. Final errors of the intrinsic parameters of synthetic point cloud #1

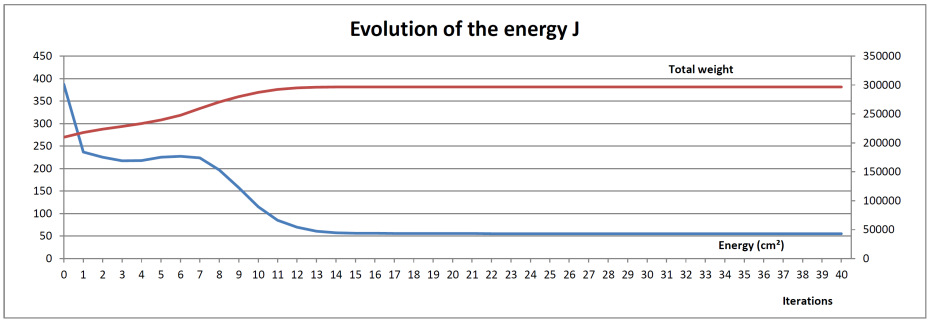

Figure 12. Evolution of the energy of the real point cloud #2. In blue, we show the evolution of our energy during the optimization; in red, this is the number of paired points at each iteration for our optimization method

	$\Delta\rho(cm^2)$	$\Delta\theta(^{\circ 2})$	$\Delta\phi(^{\circ 2})$	$\Delta H_z(cm^2)$
Initial errors	3100	193.75	279	3100
Final errors after our optimization	191.15	3.06	2.01	1343.77

Table 13. Final errors of the intrinsic parameters of real point cloud #2

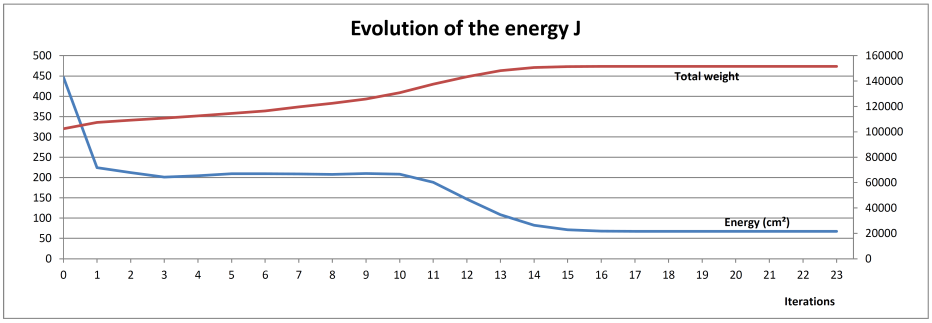

Figure 14. Evolution of the energy of the real point cloud #3. In blue, we show the evolution of our energy during the optimization; in red, this is the number of paired points at each iteration for our optimization method

	$\Delta\rho(cm^2)$	$\Delta\theta(^{\circ 2})$	$\Delta\phi(^{\circ 2})$	$\Delta H_z(cm^2)$
Initial errors	3100	193.75	279	3100
Final errors after our optimization	452.28	15.47	2.08	1352.10

Table 15. Final errors of the intrinsic parameters of real point cloud #3

ally show the improvements on the structure of the point cloud - and the expected result was smaller errors, which improved the structure of the point clouds.

Figure 8 shows the improvements on point cloud #2 with our op-

timization: on top, this is the point cloud with the added errors, and at the bottom with optimized intrinsic parameters. Figure 12 gives the evolution of the energy with the iterations for point cloud #2: we see that the energy has an important decrease, from a value of 387.39 cm^2 to a value of 55.03 cm^2, which shows that the structure of the point cloud has been improved, and which can validate the optimization result. Finally, table 13 gives the final errors for the optimized intrinsic parameters: as expected, we can see that with the optimization, we have smaller intrinsic biases.
We have the same observations for point cloud #3: figure 9 shows the improvements on the point cloud, and figure 14 the evolution of the energy with the iterations. Table 15 shows the same results as for point cloud #2.
Finally, the computation times are longer for the real point clouds, because there is more iterations for the optimization. We have respectively for point cloud #2 and #3 computation times of 25 minutes, and 12 minutes.

5. CONCLUSION

We presented in this paper a novel method for doing the automatic optimization of the intrinsic calibration parameters of a terrestrial LIDAR system, in a post-processing application. We correct the intrinsic model of a multi-beam LIDAR with an optimization problem. The optimization process we use is robust to large initial errors, as showed with the optimization results: it gives corrected calibration parameters and a well-structured point cloud, where the global noise is reduced.
Also, we presented results on real point clouds acquired by a Velodyne multi-beam sensor: our optimization can be applied to any multi-beam LIDAR sensor configuration, as long as there is overlapping data between the beams.

REFERENCES

Atanacio-jiménez, G., Hurtado-ramos, J. B. and González-barbosa, R., 2011. Lidar Velodyne HDL-64E Calibration using Pattern Planes". *International Journal of Advanced Robotic Systems* pp. 70–82.

Chan, T. O. and Lichti, D. D., 2013. "Feature-based self-calibration of Velodyne HDL-32E LiDAR for terrestrial mobile mapping applications". The 8th International Symposium on Mobile Mapping Technology, Tainan, Taiwan.

Chan, T. O. and Lichti, D. D., 2015. "Automatic in Situ Calibration of a Spinning Beam LIDAR System in Static and Kinematic Modes". *Remote Sensing* pp. 10480–10500.

Eigen library, 2015. `http://eigen.tuxfamily.org/index.php?title=Main_Page`. Last accessed: 2016-03-15.

FLANN library, 2015. `http://www.cs.ubc.ca/research/flann`. Last accessed: 2016-03-15.

Glennie, C. and Lichti, D. D., 2010. "Static Calibration and Analysis of the Velodyne HDL-64E S2 for High Accuracy Mobile Scanning". *Remote Sensing* Vol. 2(6), pp. 1610–1624.

Grand Darpa Challenge, 2007. `http://en.wikipedia.org/wiki/DARPA_Grand_Challenge`. Last accessed: 2016-03-15.

Huang, P.-S., Hong, W.-B., Chien, H.-J. and Chen, C.-Y., 2013. "Extrinsic Calibration of a multi-beam LIDAR System with improved Intrinsic Laser Parameters using V-Shaped Planes and infrared Images". Proceedings of the 11th IEEE IVMSP.

Levinson, J. and Thrun, S., 2010. "Unsupervised Calibration for Multi-beam Lasers". International Symposium on Experimental Robotics.

Lin, C.-C., Liao, Y.-D. and Luo, W.-J., 2013. "Calibration Method for Extending Single-Layer LIDAR to Multi-Layer LIDAR". Proceedings of the 2013 IEEE/SICE International Symposium on System Integration, Kobe International Conference Center, Kobe, Japan pp. 677–681.

Muhammad, N. and Lacroix, S., 2010. "Calibration of a rotating Multi-Beam LIDAR". IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan.

Narayana K. S, K., Choi, S. and Goulette, F., 2009. "Localization For Mobile Mapping Systems, Experimental Results, Analysis and Post-Processing Improvements". 6th International Symposium on Mobile Mapping Technology, Sao Paulo, Brazil.

Nouira, H., Deschaud, J. and Goulette, F., 2015. "Target-free Extrinsic Calibration of a Mobile Multi-Beam LIDAR System". ISPRS Geospatial Week, La Grande Motte, France.

Nuchter, A., Surmann, H., Lingemann, K., Hertzberg, J. and Thrun, S., 2004. "6D SLAM with an Application in Autonomous Mine Mapping". Proceedings of the IEEE International Conference on Robotics and Automation, New Orleans, L.A. (April), pp. 1998–2003.

Quanergy product page, 2015. `http://www.lidarusa.com/m8.php`. Last accessed: 2016-03-15.

Riegl LIDAR sensor datasheet, 2015. `http://www.riegl.com/uploads/tx_pxpriegldownloads/DataSheet_VMX-450_2015-03-19.pdf`. Last accessed: 2016-03-15.

Serna, A. and Marcotegui, B., 2014. "Detection, Segmentation and Classification of 3D Urban Objects using Mathematical Morphology and Supervised Learning". *ISPRS Journal of Photogrammetry and Remote Sensing* vol. 93, pp. 243–255.

Sheehan, M., Harrison, A. and Newman, P., 2012. "Self-calibration for a 3D laser". *The International Journal of Robotics Research*, vol. 31 (april), pp. 675–687.

Velodyne site web, 2015. `http://velodynelidar.com/products.html`. Last accessed: 2016-03-15.